Web Results **1 - 10** of about **101,000** for **finalizer garbage collector time reduction**. (**0.19** seconds)

Performance Tuning **Garbage Collection** in Java
0.2300771 secs - **time** it took for **garbage collection** to occur. ... finalization (**finalize** method, or finally clauses) will cause lag in **garbage collection**. ...
www.petefreitag.com/articles/gctuning/ - 26k - Cached - Similar pages

[PS] Design and Implementation of a Concurrent **Garbage Collector** for Rotor
File Format: Adobe PostScript - View as Text
able queue, is a **finalizer** thread that keeps monitoring. the freachable queue for any new entries ... **time garbage collector** based on the lifetimes of ob- ...
archive.csa.iisc.ernet.in/filt/TR/2005/2/IISc-CSA-TR-2005-2.ps.gz - Similar pages

Theocacao: **Garbage Collection** and Objective-C
If there is a **reduction** of performance by using **garbage collection**, ... a performance hit and an unpredictable **time** that is not good for some kinds of apps. ...
theocacao.com/document.page/258 - 28k - Cached - Similar pages

Improving Java Application Performance and Scalability by Reducing ...
Reducing **Garbage Collection Times**. Java applications have two types of ... since the **collector** needs to do extra processing [30] and run the **finalize** method ...
developers.sun.com/techtopics/mobility/midp/articles/**garbage**collection2/ - 206k - Cached - Similar pages

UNIX App Migration: Memory and File Management
The optimizing engine of the **garbage collector** determines the best **time** to perform a ... **Finalize** method, which allows an object to clean up its unmanaged ...
www.microsoft.com/technet/interopMigration/taskstools/migrate/unix/ucamg/ch05uav4.mspx - 57k - Cached - Similar pages

[PDF] A Comparative Evaluation of Parallel **Garbage Collector** Implementations
File Format: PDF/Adobe Acrobat - View as HTML
as roots for a mark or mark/copy phase E. **Finalize** methods are executed by a distin- ...
linear **reduction** in both major and minor **collection times**. ...
www.research.ibm.com/people/d/dfb/papers/Attanasio01Comparative.pdf - Similar pages

If broken it is, fix it you should : .NET Memory: My object is not ...
It was still alive (rooted) last **time** a **garbage collection** for that specific generation ... Or it could be a member variable of an object with a **finalizer**. ...
blogs.msdn.com/tess/archive/2006/02/02/523597.aspx - 25k - Cached - Similar pages

[Python-Dev] finalization again
This leads to (there's that weird echo again :-) Boehm's solution: Call A's **finalizer** and leave the rest to the next **time** the **garbage collection** runs. ...
mail.python.org/pipermail/python-dev/2000-March/002514.html - 10k - Cached - Similar pages

[PDF] Smarter **Garbage Collection** with Simplifiers
File Format: PDF/Adobe Acrobat
management (**garbage collection**), run-**time** environments. General Terms: Language, Performance. Keywords: **finalizer**, simplifier, lightweight daemon, ...
portal.acm.org/ft_gateway.cfm?id=1178601&

type=pdf&coll=&dl=ACM&CFID=15151515&CFTOKEN=... - Similar pages

[PDF] Modern **Garbage Collection** for Virtual Machines
File Format: PDF/Adobe Acrobat - View as HTML
A **finalizer** is asynchronously invoked by the **garbage collector** ... The **reduction** in pause **time**. over a conventional stop-the-world **collector** is based on the ...
cs.ucsb.edu/~sunils/pubs/mae.pdf - Similar pages

Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**

Web Results **1** - **10** of about **278,000** for **tuning Java garbage collection metrics**. (**0.14** seconds)

Tuning Garbage Collection with the 1.4.2 **Java**[tm] Virtual Machine
Tuning Garbage Collection with the 1.4.2 **Java**[tm] Virtual Machine ... sensitivity of your individual performance **metric** to the **garbage collector** parameters. ...
java.sun.com/docs/hotspot/gc1.4.2/ - 83k - Cached - Similar pages

Tuning Garbage Collection with the 1.3.1 **Java** Virtual Machine
Tuning Garbage Collection with the 1.3.1 **Java** Virtual Machine ... then explore the sensitivity of your individual performance **metric** to the GC parameters. ...
java.sun.com/docs/hotspot/gc/ - 28k - Cached - Similar pages
[More results from java.sun.com]

Performance **Tuning Garbage Collection** in **Java**
A summary of tips for optomizing performance of **garbage collection** in **java**. ... Then graph your own performance **metric** against young generation sizes to ...
www.petefreitag.com/articles/gc**tuning**/ - 26k - Cached - Similar pages

Garbage collection tuning in **Java** 5.0 is a breath of fresh air
Peter V. Mikhalenko offers a review of **Java** 5.0 capabilities in the **tuning** of **garbage collection**, and explains how you can minimize the impact of **garbage** ...
builder.com.com/5100-6370_14-6108296.html - 34k - Cached - Similar pages

Performance **tuning Java**™: performance tools - HP DSPP
HPjtune is a **Java Garbage Collection** (GC) visualization tool for analyzing ... User-configurable graphs for access to selected **garbage collection metrics**. ...
h21007.www2.hp.com/dspp/tech/tech_TechDocumentDetailPage_IDX/1,1701,1620,00.html - 63k - Cached - Similar pages

Garbage collection tuning in **Java** 5.0 - Program - **Java** C C++ ...
Garbage collection tuning in **Java** 5.0. By Peter V. Mikhalenko, ... sensitivity of your individual performance **metric** to the **garbage collector** parameters. ...
www.builderau.com.au/program/**java**/soa/**Garbage_collection_tuning_in_Java**_5_0/0,339024620,339269601,00.htm - 43k - Cached - Similar pages

Garbage collection tuning in **Java** 5.0 - Program - **Java** C C++ ...
There are two **metrics** of performance of a **Java** application (and **garbage collection** ... Fine-**tuning garbage collection** The command-line argument -verbose:gc ...
www.builderau.com.au/program/**java**/print.htm?TYPE=story&AT=339269601-339024620t-320000991c - 17k - Cached - Similar pages
[More results from www.builderau.com.au]

Java technology, IBM style: **Garbage collection** policies, Part 2
Use verbose GC and application **metrics** to optimize **garbage collection** ... "Fine-**tuning Java garbage collection** performance," Sumit Chawla (developerWorks, ...
www.ibm.com/developerworks/**java**/library/j-ibm**java**3/index.html - 72k - Cached - Similar pages

Tuning the **Java** Runtime System
For detail information on **Tuning Garbage Collection**, ... Accordingly, graph your own performance **metric** against young generation sizes to find the best ...

docs.sun.com/source/817-2180-10/pt_chap5.html - 24k - Cached - Similar pages

Software Secret Weapons: The Last **Java Garbage Collection** Guide ...
But before you run to your boss with the great results of your **Java garbage collection tuning**, do not forget to run the control experiments and establish ...
www.softwaresecretweapons.com/jspwiki/Wiki.jsp?page=TheLast**JavaGarbageCollection**GuideYouWillEverNeed - 43k - Cached - Similar pages

Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**

Terms used
estimating same garbage collection near4 time duration

Found **89,599** of **196,064**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 New garbage collection algorithms and strategies: Garbage-first garbage collection

David Detlefs, Christine Flood, Steve Heller, Tony Printezis
October 2004 **Proceedings of the 4th international symposium on Memory management ISMM '04**
Publisher: ACM Press

Full text available: pdf(199.59 KB) Additional Information: full citation, abstract, references, citings, index terms

Garbage-First is a server-style garbage collector, targeted for multi-processors with large memories, that meets a soft real-time goal with high probability, while achieving high throughput. Whole-heap operations, such as global marking, are performed concurrently with mutation, to prevent interruptions proportional to heap or live-data size. Concurrent marking both provides collection "completeness" and identifies regions ripe for reclamation via compacting evacuation. This ev ...

Keywords: concurrent garbrage collection, garbage collection, garbage-first garbage collection, parallel garbage collection, soft real-time garbage collection

2 Garbage collection: A true hardware read barrier

Matthias Meyer
June 2006 **Proceedings of the 2006 international symposium on Memory management ISMM '06**
Publisher: ACM Press

Full text available: pdf(991.66 KB) Additional Information: full citation, abstract, references, index terms

Read barriers synchronize compacting garbage collection and application processing in a simple yet elegant way. Unfortunately, read barrier checks are expensive to implement in software, and even with hardware support, the clustering of read barrier faults irregularly impairs application progress to an unacceptable extent. For this reason, read barriers are often considered unsuitable for hard real-time systems.In this paper, we introduce a novel hardware read barrier design for an object-based ...

Keywords: hardware support, object-based processor architecture, read barrier, real-time garbage collection

3 Real-time garbage collection for flash-memory storage systems of real-time

embedded systems
Li-Pin Chang, Tei-Wei Kuo, Shi-Wu Lo
November 2004 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 3 Issue 4
Publisher: ACM Press
Full text available: pdf(465.38 KB) Additional Information: full citation, abstract, references, citings, index terms

Flash-memory technology is becoming critical in building embedded systems applications because of its shock-resistant, power economic, and nonvolatile nature. With the recent technology breakthroughs in both capacity and reliability, flash-memory storage systems are now very popular in many types of embedded systems. However, because flash memory is a write-once and bulk-erase medium, we need a translation layer and a garbage-collection mechanism to provide applications a transparent storage ...

Keywords: Embedded systems, flash memory, garbage collection, real-time system, storage systems

4 Real-time replication garbage collection
Scott Nettles, James O'Toole
June 1993 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1993 conference on Programming language design and implementation PLDI '93**, Volume 28 Issue 6
Publisher: ACM Press
Full text available: pdf(1.05 MB) Additional Information: full citation, abstract, references, citings, index terms

We have implemented the first copying garbage collector that permits continuous unimpeded mutator access to the original objects during copying. The garbage collector incrementally replicates all accessible objects and uses a mutation log to bring the replicas up-to-date with changes made by the mutator. An experimental implementation demonstrates that the costs of using our algorithm are small and that bounded pause times of 50 milliseconds can be readily achieved.

Keywords: concurrent collection, copying garbage collection, incremental collection, real-time garbage collection, replication

5 Concurrent garbage collection using hardware-assisted profiling
Timothy H. Heil, James E. Smith
October 2000 **ACM SIGPLAN Notices , Proceedings of the 2nd international symposium on Memory management ISMM '00**, Volume 36 Issue 1
Publisher: ACM Press
Full text available: pdf(1.74 MB) Additional Information: full citation, abstract, citings, index terms

In the presence of on-chip multithreading, a Virtual Machine (VM) implementation can readily take advantage of *service threads* for enhancing performance by performing tasks such as profile collection and analysis, dynamic optimization, and garbage collection concurrently with program execution. In this context, a hardware-assisted profiling mechanism is proposed. The *Relational Profiling Architecture* (RPA) is designed from the top down. RPA is based on a relational model similar ...

6 Garbage collection and local variable type-precision and liveness in Java virtual machines
Ole Agesen, David Detlefs, J. Eliot Moss
May 1998 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1998 conference on Programming language design and implementation PLDI '98**, Volume 33

Issue 5
Publisher: ACM Press

Full text available: pdf(1.54 MB) Additional Information: full citation, abstract, references, citings, index terms

Full precision in garbage collection implies retaining only those heap allocated objects that will actually be used in the future. Since full precision is not computable in general, garbage collectors use safe (i.e., conservative) approximations such as reachability from a set of root references. Ambiguous roots collectors (commonly called "conservative") can be overly conservative because they overestimate the root set, and thereby retain unexpectedly large amounts of garbage. We consider two m ...

7 Older-first garbage collection in practice: evaluation in a Java Virtual Machine

Darko Stefanović, Matthew Hertz, Stephen M. Blackburn, Kathryn S. McKinley, J. Eliot B. Moss
June 2002 **ACM SIGPLAN Notices , Proceedings of the 2002 workshop on Memory system performance MSP '02**, Volume 38 Issue 2 supplement
Publisher: ACM Press
Full text available: pdf(1.15 MB) Additional Information: full citation, abstract, references, citings

Until recently, the best performing copying garbage collectors used a generational policy which repeatedly collects the very youngest objects, copies any survivors to an older space, and then infrequently collects the older space. A previous study that used garbage-collection simulation pointed to potential improvements by using an *Older-First* copying garbage collection algorithm. The Older-First algorithm sweeps a fixed-sized window through the heap from older to younger objects, and avo ...

8 List processing in real time on a serial computer

Henry G. Baker
April 1978 **Communications of the ACM**, Volume 21 Issue 4
Publisher: ACM Press
Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, references, citings, index terms

A real-time list processing system is one in which the time required by the elementary list operations (e.g. CONS, CAR, CDR, RPLACA, RPLACD, EQ, and ATOM in LISP) is bounded by a (small) constant. Classical implementations of list processing systems lack this property because allocating a list cell from the heap may cause a garbage collection, which process requires time proportional to the heap size to finish. A real-time list processing system is presented which continuously reclaims garb ...

Keywords: CDR-coding, LISP, compacting, file or database management, garbage collection, list processing, real-time, reference counting, storage allocation, storage management, virtual memory

9 Tuning garbage collection for reducing memory system energy in an embedded java environment

G. Chen, R. Shetty, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, M. Wolczko
November 2002 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 1 Issue 1
Publisher: ACM Press
Full text available: pdf(740.23 KB) Additional Information: full citation, abstract, references, citings, index terms

Java has been widely adopted as one of the software platforms for the seamless integration of diverse computing devices. Over the last year, there has been great momentum in adopting Java technology in devices such as cellphones, PDAs, and pagers

where optimizing energy consumption is critical. Since, traditionally, the Java virtual machine (JVM), the cornerstone of Java technology, is tuned for performance, taking into account energy consumption requires reevaluation, and possibly redesign of t ...

Keywords: Garbage collector, Java Virtual Machine (JVM), K Virtual Machine (KVM), low power computing

10 An implementation of complete, asynchronous, distributed garbage collection

Fabrice Le Fessant, Ian Piumarta, Marc Shapiro

May 1998 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1998 conference on Programming language design and implementation PLDI '98**, Volume 33 Issue 5

Publisher: ACM Press

Full text available: pdf(1.20 MB) Additional Information: full citation, abstract, references, citings, index terms

Most existing reference-based distributed object systems include some kind of acyclic garbage collection, but fail to provide acceptable collection of cyclic garbage. Those that do provide such GC currently suffer from one or more problems: synchronous operation, the need for expensive global consensus or termination algorithms, susceptibility to communication problems, or an algorithm that does not scale. We present a simple, complete, fault-tolerant, asynchronous extension to the (acyclic) cle ...

Keywords: distributed object systems, garbage collection, reference tracking, storage management

11 The KaffeOS Java runtime system

Godmar Back, Wilson C. Hsieh

July 2005 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 27 Issue 4

Publisher: ACM Press

Full text available: pdf(704.30 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Single-language runtime systems, in the form of Java virtual machines, are widely deployed platforms for executing untrusted mobile code. These runtimes provide some of the features that operating systems provide: interapplication memory protection and basic system services. They do not, however, provide the ability to isolate applications from each other. Neither do they provide the ability to limit the resource consumption of applications. Consequently, the performance of current systems degra ...

Keywords: Robustness, garbage collection, isolation, language runtimes, resource management, termination, virtual machines

12 Formal semantics and static analysis: Compiler optimizations for nondeferred reference: counting garbage collection

Pramod G. Joisha

June 2006 **Proceedings of the 2006 international symposium on Memory management ISMM '06**

Publisher: ACM Press

Full text available: pdf(220.00 KB) Additional Information: full citation, abstract, references, index terms

Reference counting is a well-known technique for automatic memory management, offering unique advantages over other forms of garbage collection. However, on account of the high costs associated with the maintenance of up-to-date tallies of references from the

stack, deferred variants are typically used in modern implementations. This partially sacrifices some of the benefits of non-deferred reference-counting (RC) garbage collection, like the immediate reclamation of garbage and short collector ...

Keywords: reference counting, static analyses

13 An experimental study of renewal-older-first garbage collection

Lars T. Hansen, William D. Clinger
September 2002 **ACM SIGPLAN Notices , Proceedings of the seventh ACM SIGPLAN international conference on Functional programming ICFP '02**, Volume 37 Issue 9
Publisher: ACM Press

Full text available: pdf(143.87 KB) Additional Information: full citation, abstract, references, citings, index terms

Generational collection has improved the efficiency of garbage collection in fast-allocating programs by focusing on collecting young garbage, but has done little to reduce the cost of collecting a heap containing large amounts of older data. A new generational technique, older-first collection, shows promise in its ability to manage older data.This paper reports on an implementation study that compared two older-first collectors to traditional (younger-first) generational collectors. One of the ...

Keywords: generational garbage collection, older-first

14 Performance analysis of on-the-fly garbage collection

Tim Hickey, Jacques Cohen
November 1984 **Communications of the ACM**, Volume 27 Issue 11
Publisher: ACM Press
Full text available: pdf(828.76 KB) Additional Information: full citation, references, citings, index terms

Keywords: efficiency, list processing, marking algorithms, parallel garbage collection, speedup

15 Objects and their collection: The pauseless GC algorithm

Cliff Click, Gil Tene, Michael Wolf
June 2005 **Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments VEE '05**
Publisher: ACM Press

Full text available: pdf(440.91 KB) Additional Information: full citation, abstract, references, citings, index terms

Modern transactional response-time sensitive applications have run into practical limits on the size of garbage collected heaps. The heap can only grow until GC pauses exceed the response-time limits. Sustainable, scalable concurrent collection has become a feature worth paying for.Azul Systems has built a custom system (CPU, chip, board, and OS) specifically to run garbage collected virtual machines. The custom CPU includes a read barrier instruction. The read barrier enables a highly concurren ...

Keywords: Java, concurrent GC, custom hardware, garbage collection, memory management, read barriers

16 Extended ephemeral logging: log storage management for applications with long lived transactions

John S. Keen, William J. Dally

March 1997 **ACM Transactions on Database Systems (TODS)**, Volume 22 Issue 1

Publisher: ACM Press

Full text available: pdf(566.34 KB) Additional Information: full citation, references, index terms, review

Keywords: OLTP, disk management, logging, long transactions

17 Practical byzantine fault tolerance and proactive recovery

Miguel Castro, Barbara Liskov

November 2002 **ACM Transactions on Computer Systems (TOCS)**, Volume 20 Issue 4

Publisher: ACM Press

Full text available: pdf(1.63 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Our growing reliance on online services accessible on the Internet demands highly available systems that provide correct service without interruptions. Software bugs, operator mistakes, and malicious attacks are a major cause of service interruptions and they can cause arbitrary behavior, that is, Byzantine faults. This article describes a new replication algorithm, BFT, that can be used to build highly available systems that tolerate Byzantine faults. BFT can be used in practice to implement re ...

Keywords: Byzantine fault tolerance, asynchronous systems, proactive recovery, state machine replication, state transfer

18 Design and evaluation of dynamic optimizations for a Java just-in-time compiler

Toshio Suganuma, Toshiaki Yasue, Motohiro Kawahito, Hideaki Komatsu, Toshio Nakatani

July 2005 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 27 Issue 4

Publisher: ACM Press

Full text available: pdf(1.60 MB) Additional Information: full citation, abstract, references, citings, index terms

The high performance implementation of Java Virtual Machines (JVM) and Just-In-Time (JIT) compilers is directed toward employing a dynamic compilation system on the basis of online runtime profile information. The trade-off between the compilation overhead and performance benefit is a crucial issue for such a system. This article describes the design and implementation of a dynamic optimization framework in a production-level Java JIT compiler, together with two techniques for profile-directed o ...

Keywords: JIT compiler, Recompilation, adaptive optimization, code specialization, dynamic compilation, profile-directed method inlining

19 Toward real-time performance benchmarks for Ada

Russell M. Clapp, Louis Duchesneau, Richard A. Volz, Trevor N. Mudge, Timothy Schultze

August 1986 **Communications of the ACM**, Volume 29 Issue 8

Publisher: ACM Press

Full text available: pdf(2.17 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Benchmarks are developed to measure the Ada notion of time, the Ada features believed

important to real-time performance, and other time-related features that are not part of the language, but are part of the run-time system; these benchmarks are then applied to the language and run-time system, and the results evaluated.

20 A memory-efficient real-time non-copying garbage collector

Tian F. Lim, Przemysław Pardyak, Brian N. Bershad
October 1998 **ACM SIGPLAN Notices , Proceedings of the 1st international symposium on Memory management ISMM '98**, Volume 34 Issue 3
Publisher: ACM Press

Full text available: pdf(1.58 MB) Additional Information: full citation, abstract, references, citings, index terms

Garbage collectors used in embedded systems such as Personal Java and Inferno or in operating systems such as SPIN must operate with limited resources and minimize their impact on application performance. Consequently, they must maintain short real-time pauses, low overhead, and a small memory footprint. Most garbage collectors, including the Treadmill algorithm, are inadequate because they sacrifice space for time. We have implemented a new Treadmill variant that provides good memory utilizatio ...

Keywords: garbage collection, operating systems, real-time, treadmill

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next